

**METHOD AND MECHANISM OF ACCESSING SEGMENTS IN DATA STORAGE  
SYSTEMS**

**Inventors:**

**Jonathan D. Klein  
Redwood City, California  
Citizenship: U.S.A.**

**Amit Ganesh  
San Jose, California  
Citizenship: India**

**Assignee:**

**Oracle International Corporation  
500 Oracle Parkway  
Redwood Shores, California 94065**

**Prepared By:**

**Peter C. Mei  
Bingham McCutchen LLP  
Three Embarcadero Center, Suite 1800  
San Francisco, California 94111  
(650) 849-4870**

**Express Mail Label No. EV348163933US**

## SPECIFICATION

### METHOD AND MECHANISM OF ACCESSING SEGMENTS IN DATA STORAGE SYSTEMS

#### BACKGROUND AND SUMMARY

5        [0001]     The present invention is related to data storage systems. More particularly, the present invention is directed to a method and mechanism of accessing segments in data storage systems.

         [0002]     Conventional data storage systems include one or more storage devices connected to a controller or manager. As used herein, the term "data storage device" refers to  
10 any device or apparatus utilizable for the storage of data, e.g., a disk drive. For explanatory purposes only and not as an intent to limit the scope of the invention, the term "disk drive" as used in this document is synonymous with the term "data storage device."

         [0003]     Data storage devices typically store data on one or more magnetic discs called platters. To access stored data, the platter on which the data is stored has to be moved into the  
15 correct position before the data can be read, i.e., the sector of the platter where the data is located on has to be positioned under a special electromagnetic read/write device called a head, which is mounted onto a slider on an arm that is positioned over the surface of the platter by an actuator. Once the platter is in the correct position, the platter rotates at high speed, driven by a spindle motor connected to the spindle that the platter is mounted on, and the head reads the  
20 data on the platter as it flies by.

         [0004]     The time it takes to move the platter into the correct position, i.e., the seek time, is frequently the most expensive part of an I/O (input/output) operation. The seek time can vary anywhere from 5-15 ms (milliseconds). Whereas the time it takes to read the data from the platter once it is in the correct position could be as fast as a couple of milliseconds depending

upon the transfer rate of the disk drive, which is a function of the RPM (rotations per minute) of the disk drive, and the amount of data to be read.

[0005] Data in data storage systems are usually organized into rows, blocks, extents, segments, and tables. Data are stored in rows. Rows, in turn, are stored in blocks. Each block, also referred to as a page, corresponds to a specific number of bytes of physical space on a data storage device. The size of a block is usually set at a prescribed level when the data storage system is created and will typically be a multiple of the operating system's block size.

[0006] Blocks, in turn, are stored in extents. An extent is a contiguous set of blocks on a disk drive. Extents are stored in segments, which are logical containers each holding a set of extents. For various reasons, e.g., load balancing, random access, etc., extents of a segment may not be contiguous on a data storage device, may span multiple files on a data storage device, and may even span multiple data storage devices. Each table comprises one or more segments.

[0007] There are two types of data access in data storage systems: random and sequential. An example of random access is access using RowIDs. A RowID is a physical identifier of a row. RowIDs may be stored separately from the table, e.g., in an index. A full table scan is an example of sequential access. In a full table scan, the one or more segments in the table have to be identified and all of the extents in each segment have to be located. I/O operations are then submitted to access each segment. The I/O operations are usually submitted on an extent by extent basis since extents are contiguous blocks on a data storage device and extents in each segment may be striped across multiple data storage devices.

[0008] Given that the I/O operations are submitted on an extent by extent basis, performance of full table scans can vary greatly depending upon the number of extents in the segment(s) of each table. Fewer extents mean less I/O operations, which result in less time spent seeking data and more time spent reading data. In order to optimize space utilization, however, extent sizes in data storage systems are usually minimized. Smaller extent sizes

Express Mail No. EV 348163933 US

Patent  
OI7034312001

generally lead to an increase in the number of extents per segment as more extents will be needed to store the same amount of data.

5 [0009] Accordingly, it is desirable to provide a method and mechanism where the performance of a full table scan is unaffected by the size of the extents in the segment(s) of the table.

[0010] The present invention provides a method and mechanism of accessing segments in data storage systems. In one embodiment, a plurality of extents in a segment are coalesced into a plurality of groups based on data storage device location. A single I/O operation is then submitted for each group of extents.

10 [0011] Further details of aspects, objects, and advantages of the invention are described below in the detailed description, drawings, and claims. Both the foregoing general description and the following detailed description are exemplary and explanatory, and are not intended to be limiting as to the scope of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] The accompanying drawings are included to provide a further understanding of the invention and, together with the Detailed Description, serve to explain the principles of the invention.

5 [0013] Fig. 1 is a flow chart of a method of accessing segments in data storage systems according to one embodiment of the invention.

[0014] Figs. 2-3 illustrate examples of segments according to various embodiments of the invention.

10 [0015] Fig. 4 depicts a process flow of a method of accessing segments in data storage systems according to another embodiment of the invention.

[0016] Fig. 5 shows a sample segment according to an embodiment of the invention.

[0017] Figs. 6-7 are different embodiments of methods of accessing segments in data storage systems.

15 [0018] Fig. 8 illustrates a diagram of a computer system with which embodiments of the present invention can be implemented.

DETAILED DESCRIPTION

[0019] Access of segments in data storage systems is disclosed. Rather than submit an I/O operation for each extent in a segment, which can dramatically affect performance of sequential access in data storage systems, some or all of the extents in the segment are grouped  
5 together based on data storage device location and a single I/O operation is submitted for each group of extents. This reduces the effects that smaller extent sizes have on the performance of full table scans.

[0020] Illustrated in Fig. 1 is a method of accessing segments in data storage systems according to one embodiment of the invention. At 102, a plurality of extents in a segment is  
10 coalesced into a plurality of groups based on data storage device location. A single I/O operation is then submitted for each group of extents at 104.

[0021] Fig. 2 depicts a segment 200 comprising extents 202-220. Extents 202-220 are logically numbered from 1 to 10. The extents in segment 200 are striped over three data storage devices 222-226. As seen from the example in Fig. 2, although extents 202, 208, 214, and 220 are  
15 not logically contiguous, they are physically contiguous on data storage device 222. In one embodiment, all of the extents in segment 200 are coalesced into three groups based on data storage device location, i.e., extents 202, 208, 214, and 220 form one group, extents 204, 210, and 216 form the another group, and extents 206, 212, and 218 form the last group. To access  
20 segment 200, one I/O operation is submitted for each group of the extents. Hence, instead of submitting ten I/O operations, which can greatly affect the performance of a full scan of a table comprising segment 200, only three I/O operations need to be submitted.

[0022] An I/O operation may return one or more extents that are not in the segment being accessed. As shown in Fig. 3, a segment 300 comprises extents 302-314, which are logically numbered 1 to 7. Extents 302-314 are striped over two data storage devices 326-328.  
25 Although extents 302, 306, 310, and 314 are all stored on disk 326, extent 302 is not contiguous

Express Mail No. EV 348163933 US

Patent  
OI7034312001

with extents 306, 310, and 314. Two other extents 316-318, which are not part of segment 300, are sandwiched between extents 302 and 306. Thus, if one I/O is submitted for extents 302, 306, 310, and 314, the I/O will also return extents 316-318. Since memory is inexpensive, it may still be more efficient to submit one I/O for extents 302, 306, 310, and 314 instead of four I/Os, one for each extent.

[0023] Fig. 4 illustrates a method of accessing segments in data storage systems according to another embodiment of the invention. A plurality of extents in a segment is coalesced into a plurality of groups based on data storage device location (402). At least one of the plurality of groups is partitioned into two or more groups (404). A single I/O operation is then submitted for each group of extents (406).

[0024] A sample segment 500 is depicted in Fig. 5. Segment 500 comprises eighty extents 501-580 spread out over nine disk drives 591-599. The number of extents coalesced may be less than all of the extents in the segment. For example, if a user decided that no more than 20 extents should be coalesced at any one time, extents 501-520 would be coalesced into nine groups and an I/O operation would be submitted for each group. Extents 521-540 would be then be coalesced into seven groups and an I/O operation would be submitted for each group, and so forth.

[0025] Users may terminate access to a segment before all of the extents in the segment have been returned or even coalesced. For instance, after viewing an initial set of data, a user may decide that no other data is necessary and choose to terminate access to the segment before all of the data have been returned.

[0026] One or more coalesced groups may be further divided into additional groups based on various factors, e.g., size, performance, etc. For example, if one I/O operation is submitted for the group of extents on disk drive 591, the I/O operation would return three extents 581 and 587-588 that are not within segment 500. To improve performance, the group

Express Mail No. EV 348163933 US

Patent  
OI7034312001

may be divided into three groups, one group containing extents 501-502, another group containing extents 536-540, and the last group containing extents 562-563 and 577. Three I/O operations can then be submitted, one for each group of extents, which will exclude the three non-segment 500 extents 581 and 587-588 sandwiched between the three groups.

5        [0027]        In the examples of Figs. 2-3 and 5, only the extents in segments 200, 300, and 500 and any extents sandwiched between the extents of those segments are shown as being stored in data storage devices 222-226, 326-328, and 591-599. Data storage devices 222-226, 326-328, and 591-599, however, may contain other extents that are not shown. Additionally, although the extents in Figs. 2-3 and 5 are illustrated to be of the same size, extents may vary in size  
10        within and across segments.

         [0028]        Another method of accessing segments in data storage systems is shown in Fig. 6. At 602, a plurality of extents in a segment are coalesced into a plurality of groups based on data storage device location. A single I/O operation is submitted for each group of extents (604). The I/O operations are scheduled to overlap with CPU (central processing unit) activity (606).  
15        In one embodiment, the I/O operations are scheduled to ensure that the CPU is not idle while an I/O operation is executing.

         [0029]        By overlapping I/O operations with CPU activity, the rate in which results are returned to users, i.e., throughput, can be greatly improved. For example, after an I/O operation for a group of extents have completed, the CPU will process the returned extents, e.g.,  
20        evaluate rows in the extents against a predicate. Rather than waiting until the CPU has finished processing the first group of extents before submitting another I/O operation, an I/O operation can be submitted for another group of extents while the CPU is processing the first group of extents. If the I/O operations are scheduled correctly, when the CPU finishes processing the first group of extents, the CPU can begin processing the other group of extents as the I/O  
25        operation for the other group should have completed by then. This way, the CPU does not



Express Mail No. EV 348163933 US

Patent  
OI7034312001

remain idle after it finishes processing the first group of extents and before the I/O operation for the other group has ended.

[0030] The scheduling of I/O operations to overlap with CPU activity can also influence whether to partition any of the coalesced group of extents. For instance, if one group of extents is much larger than another group of extents, then the time it takes for the I/O operation of the larger group of extents to complete will likely be much longer than the time it takes to process the extents in the smaller group. Thus, even though the I/O operation is scheduled to overlap with CPU activity, the CPU will remain idle while it waits for the I/O operation of the larger group to finish. To better utilize resources and improve throughput, the larger group of extents can be divided into smaller groups in order to reduce the I/O operation time. Although this will increase the number of I/O operations, it may still be a more efficient use of resources and lead to improved throughput.

[0031] Depicted in Fig. 7 is a method of accessing segments in data storage systems according to an embodiment of the invention. A plurality of extents in a segment are coalesced into a plurality of groups based on data storage device location (702). A single I/O operation is submitted for each group of extents (704). The I/O operations are scheduled to overlap with CPU activity (706). Information on CPU activity and I/O operations are collected (708) and the scheduling of the I/O operations is adjusted based on the collected information (710).

[0032] The information collected may include information on whether the CPU has been idle during the last I/O operation. For example, the scheduling of I/O operations may be based upon system estimates on the amount of time it will take to process the extents returned from each of the I/O operations. Estimation of CPU activity may be incorrect due to erroneous approximation of the number of rows in each block of an extent. For instance, if the system expected thousands of rows in an extent, but it ended up processing only a few hundred rows, the CPU activity will likely end before the I/O operation scheduled to be execute during the CPU activity completes. As a result, the CPU will remain idle while it waits for the pending I/O

operation to complete. Thus, using this information, scheduling of the I/O operations can be adjusted. In addition, the partitioning of extents can be dynamically changed, e.g., one or more of the groups can be partitioned into smaller groups.

## SYSTEM ARCHITECTURE OVERVIEW

5        [0033]        Fig. 8 is a block diagram of a computer system 800 suitable for implementing an embodiment of the present invention. Computer system 800 includes a bus 802 or other communication mechanism for communicating information, which interconnects subsystems and devices, such as processor 804, system memory 806 (e.g., RAM), static storage device 808 (e.g., ROM), disk drive 810 (e.g., magnetic or optical), communication interface 812 (e.g., modem or ethernet card), display 814 (e.g., CRT or LCD), input device 816 (e.g., keyboard), and cursor control 818 (e.g., mouse or trackball).

         [0034]        According to one embodiment of the invention, computer system 800 performs specific operations by processor 804 executing one or more sequences of one or more instructions contained in system memory 806. Such instructions may be read into system memory 806 from another computer readable medium, such as static storage device 808 or disk drive 810. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement the invention.

         [0035]        The term "computer readable medium" as used herein refers to any medium that participates in providing instructions to processor 804 for execution. Such a medium may take many forms, including but not limited to, non-volatile media, volatile media, and transmission media. Non-volatile media includes, for example, optical or magnetic disks, such as disk drive 810. Volatile media includes dynamic memory, such as system memory 806. Transmission media includes coaxial cables, copper wire, and fiber optics, including wires that comprise bus 802. Transmission media can also take the form of acoustic or light waves, such as those generated during radio wave and infrared data communications.

Express Mail No. EV 348163933 US

Patent  
OI7034312001

[0036] Common forms of computer readable media includes, for example, floppy disk, flexible disk, hard disk, magnetic tape, any other magnetic medium, CD-ROM, any other optical medium, punch cards, paper tape, any other physical medium with patterns of holes, RAM, PROM, EPROM, FLASH-EPROM, any other memory chip or cartridge, carrier wave, or any  
5 other medium from which a computer can read.

[0037] In an embodiment of the invention, execution of the sequences of instructions to practice the invention is performed by a single computer system 800. According to other embodiments of the invention, two or more computer systems 800 coupled by communication link 820 (e.g., LAN, PTSN, or wireless network) may perform the sequence of instructions  
10 required to practice the invention in coordination with one another.

[0038] Computer system 800 may transmit and receive messages, data, and instructions, including program, i.e., application code, through communication link 820 and communication interface 812. Received program code may be executed by processor 804 as it is received, and/or stored in disk drive 810, or other non-volatile storage for later execution.

[0039] In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. For example, the above-described process flows are described with reference to a particular ordering of process actions. However, the ordering of many of the described process  
15 actions may be changed without affecting the scope or operation of the invention. The  
20 specification and drawings are, accordingly, to be regarded in an illustrative rather than restrictive sense.